**Design Review**

1. ***Coupling***
   1. Content coupling

|  |  |
| --- | --- |
| Related modules | Description |
|  |  |

* 1. Common coupling

|  |  |
| --- | --- |
| Related modules | Description |
| Configs and class that import Configs | Class that have coupling with Configs usually access directly to its global data |

* 1. Control coupling

|  |  |
| --- | --- |
| Related modules | Description |
| PaymentResultScreenHandler and PaymentScreenHandler | PaymentResultScreenHandler break the nested loop inside PaymentScreenHandler after user confirm the payment result so that PaymentScreenHandler finally return the result of the transaction for other class to continue their operation. |

* 1. Stamp coupling

|  |  |
| --- | --- |
| Related modules | Description |
|  |  |

* 1. Data coupling

|  |  |
| --- | --- |
| Related modules | Description |
| All other modules that have coupling with each other | Classes that communicate to each other using parameters and each parameter used is necessary |

1. ***Cohesion***
   1. Coincidental cohesion

|  |  |
| --- | --- |
| Related modules | Description |
| Utils | Methods in Utils class have no relation other than being there |

* 1. Logical cohesion

|  |  |
| --- | --- |
| Related modules | Description |
| Validator | Validator have two methods that are related only by logically but not functionally |
| Boundary classes | These classes are all GUI handler class that operate on their respective screen based on user’s interaction. The operation are varied and only related logically |
| ViewStationController | The methods in this class only related logically since they are all to provide and validate information in view station process |

* 1. Temporal cohesion

|  |  |
| --- | --- |
| Related modules | Description |
|  |  |

* 1. Procedural cohesion

|  |  |
| --- | --- |
| Related modules | Description |
| TransactionController | The methods in TransactionController follow a sequence of execution: checkTransactionInfo -> makeTransaction |
| RentBikeController | The methods in RentBikeController follow a sequence of execution: requestToRentBike -> validateBarcode (incase that requestToRentBike use barcode) -> rentBike |
| ReturnBikeController | The methods in ReturnBikeController follow a sequence of execution: requestToReturnBike -> returnBike |
| InterbankBoundary | The methods in InterbankBoundary follows a sequence of execution: allowMethods -> setUpConnection -> post |

* 1. Communicational cohesion

|  |  |
| --- | --- |
| Related modules | Description |
| Timer | All methods inside this class operate on the same data which is the timer used to count bike’s rent time |

* 1. Sequential cohesion

|  |  |
| --- | --- |
| Related modules | Description |
|  |  |

* 1. Functional cohesion

|  |  |
| --- | --- |
| Related modules | Description |
| InterbankSubsystem, InterbankSubsystemControl | These classes components serves for one specific purpose: make payment transaction |
| Bike, NormalBike, EBike, TwinBike, Invoice, Station, CreditCard, PaymentTransaction, Request, Response | Entities classes with sole purpose of represent that entity |
| Database | This class represent the database of the project and return the connection to the database |
| Configs | This class hold the information of the configurations of the software |
| BaseController, HomeController, ViewBikeController | These class have only one or even no method that dedicated for a sole purpose |